



PROJECT WORK REPORT

NAME: <FIZZA ZEESHAN>

PROGRAM: <TBSE2, 2020>

TABLE OF CONTENTS

1. Description of database	3
2. ER MODEL.....	5
3. Implementation Diagram.....	6
4. EXAMPLE SQL STATEMENTS.....	7
5. CONCLUSION	22
6. References	22
7. Appendix	22

1. DESCRIPTION OF DATABASE

- I have created an E-Commerce database. E-Commerce is when a consumer usually buys or sells a product usually over the internet. This database stores information about things such as customer data, the orders, products ordered and the shipping details.
- I have five tables in this Database which are as follows:

Shipping_details
Orders
Customer
Product
Orders_has_product

- The following tables have these attributes

Shipping_details:

- Shippingdetails_id INT (PK)
- Description VARCHAR(45)
- Shipping_status VARCHAR(45)
- Address VARCHAR(45)
- ORDER_ORDER_ID INT (FK)

Orders:

- Order_id INT (PK)
- Order_type VARCHAR(45)
- Order_date DATE
- Price INT
- Customer_customer_id (FK)

Customer:

- Customer_id INT (PK)
- Customer_name VARCHAR(45)
- Email_address VARCHAR(45)
- Phone_number INT

Product:

- Product_id INT (PK)
- Name VARCHAR(45)
- Quantity INT
- Weight INT
- Orders_order_id INT

Orders_has_product:

- Orders_order_id INT
- Product_product_id INT

- I have one to one, one to many and many to many relationships between tables in my database.

- One to one – Shipping Details and Orders

It is one to one because every single order will have the same shipping details. If one customer has an order containing 5 products. That whole order will have the same shipping details.

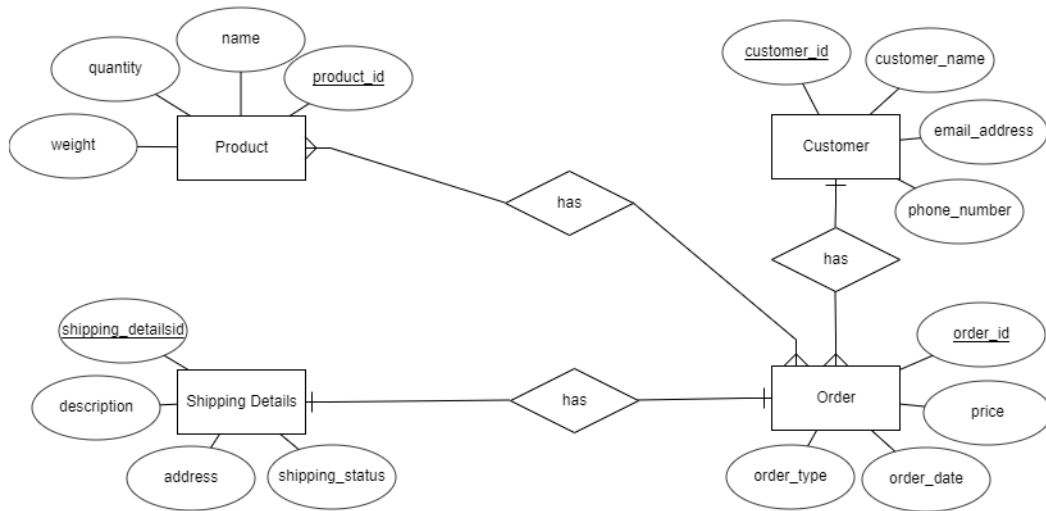
- One to many – Customer and Orders

This is one to many since one customer can place multiple orders.

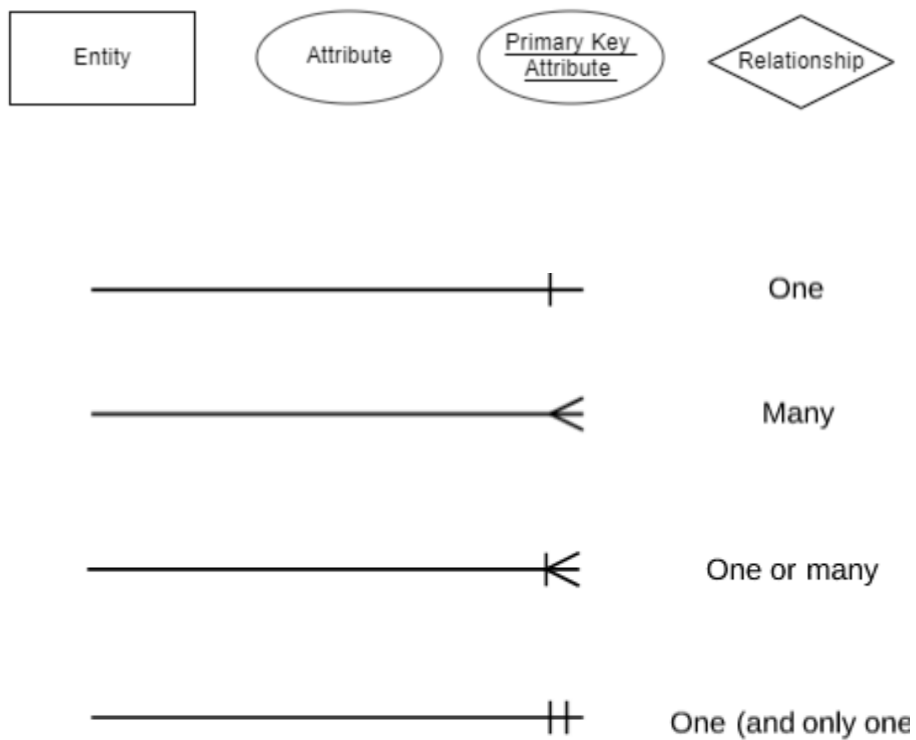
- Many to many – Product and Orders

This is many to many as an order can have many products and many products can belong to one order.

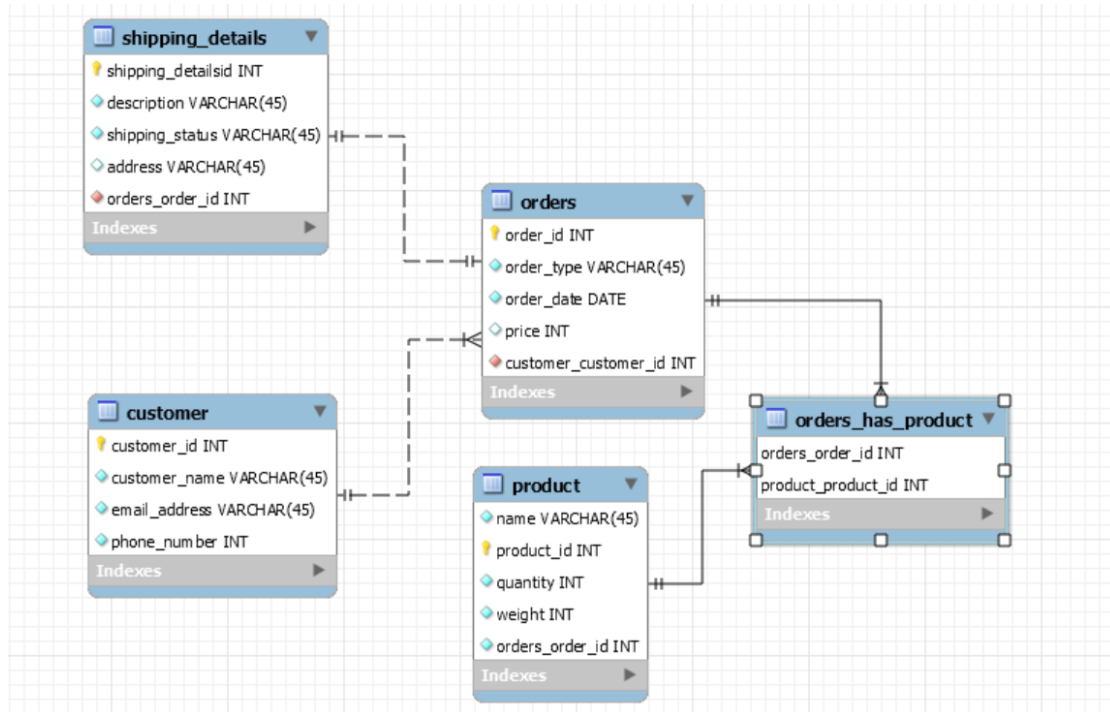
2. ER MODEL



LEGEND



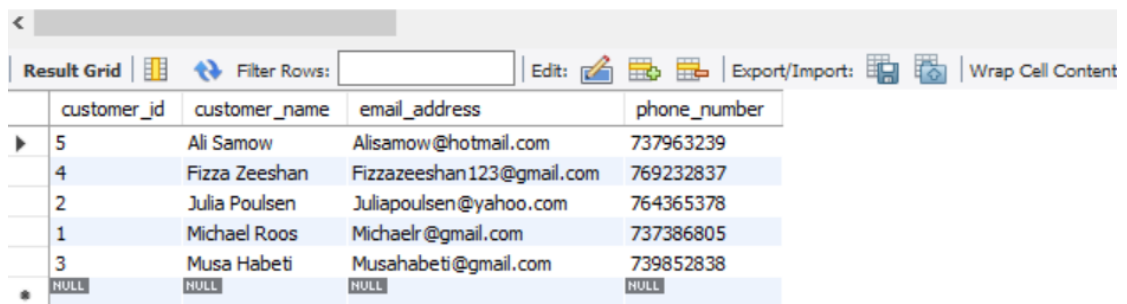
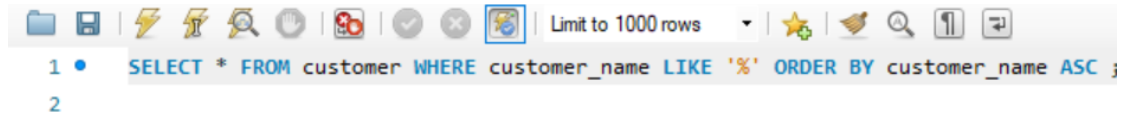
3. IMPLEMENTATION DIAGRAM



4. EXAMPLE SQL STATEMENTS

SELECT

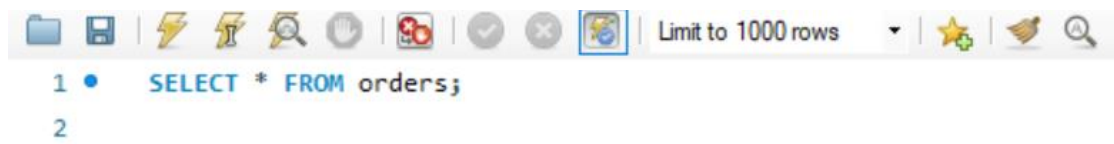
SELECT * FROM customer WHERE customer_name LIKE '%' ORDER BY customer_name ASC;



A screenshot of a database result grid showing the output of the SQL query. The grid has columns for customer_id, customer_name, email_address, and phone_number. The data is as follows:

	customer_id	customer_name	email_address	phone_number
▶	5	Ali Samow	Alisamow@hotmail.com	737963239
	4	Fizza Zeeshan	Fizzazeeshan123@gmail.com	769232837
	2	Julia Poulsen	Juliapoulsen@yahoo.com	764365378
	1	Michael Roos	Michaelr@gmail.com	737386805
	3	Musa Habeti	Musahabeti@gmail.com	739852838
*	NULL	NULL	NULL	NULL

SELECT * FROM orders;



A screenshot of a SQL query result grid. The grid displays 6 rows of data. The first five rows represent orders with details like order_id, order_type, order_date, price, and customer_customer_id. The sixth row is a summary row with NULL values. The interface includes a toolbar with icons for filtering, editing, and exporting/importing data.

	order_id	order_type	order_date	price	customer_customer_id
▶	1	Bakery Items	2021-02-22	250	1
	2	Household Items	2021-01-19	350	2
	3	Clothing	2021-01-02	800	3
	4	Outdoor	2021-02-07	1000	4
	5	Cosmetics	2021-01-11	750	5
*	NULL	NULL	NULL	NULL	NULL

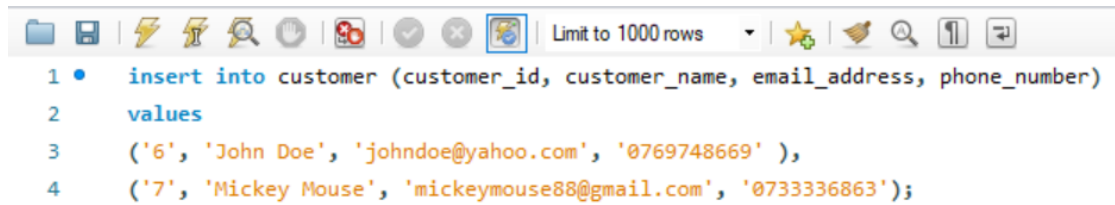
INSERT

INSERT INTO customer (customer_id , customer_name , email_address,
phone_number)

VALUES

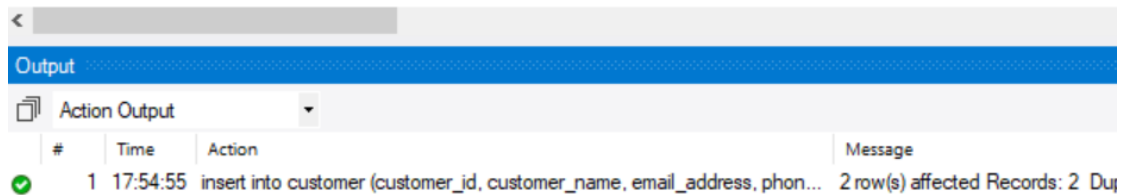
('6' , 'John Doe' , 'johndoe@yahoo.com' , '0769748669'),

('7', 'Mickey Mouse' , 'mickeymouse88@gmail.com' , '0733336863');



A screenshot of a SQL IDE interface. The top toolbar includes icons for file operations, execution, and search. A dropdown menu shows 'Limit to 1000 rows'. The main text area contains the following SQL code:

```
1 • insert into customer (customer_id, customer_name, email_address, phone_number)
2 values
3 ('6', 'John Doe', 'johndoe@yahoo.com', '0769748669' ),
4 ('7', 'Mickey Mouse', 'mickeymouse88@gmail.com', '0733336863');
```



A screenshot of the 'Output' window in the SQL IDE. It shows a table with the following data:

#	Time	Action	Message
✓ 1	17:54:55	insert into customer (customer_id, customer_name, email_address, phon...	2 row(s) affected Records: 2 Du

Limit to 1000 rows

1 • **SELECT * FROM** ecommerce.customer;

Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Cont

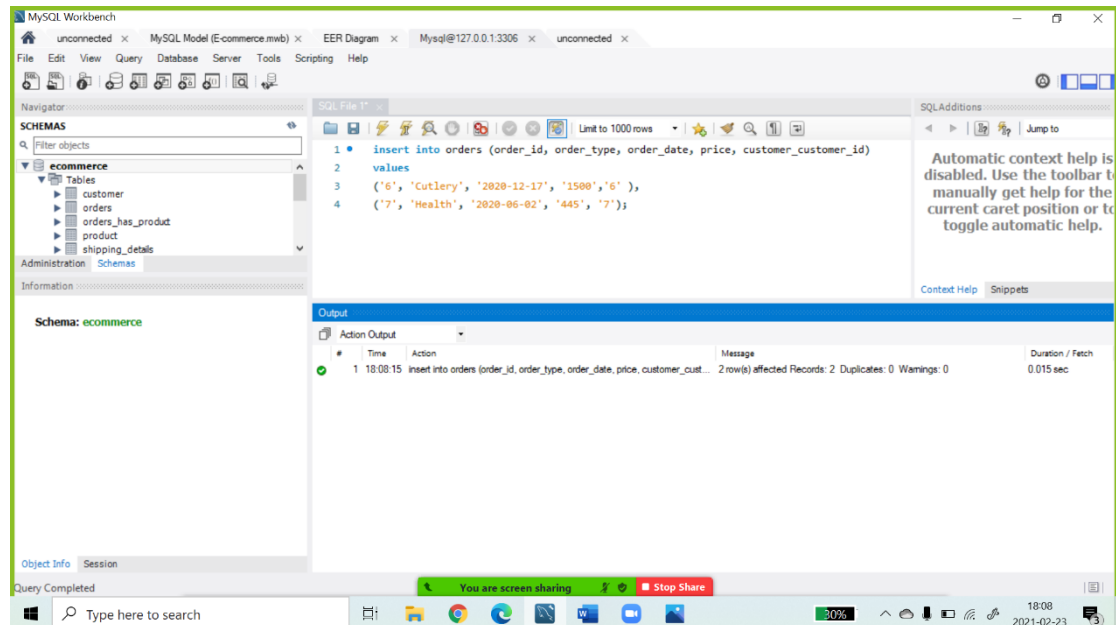
	customer_id	customer_name	email_address	phone_number
▶	1	Michael Roos	Michaelr@gmail.com	737386805
	2	Julia Poulsen	Juliapoulsen@yahoo.com	764365378
	3	Musa Habeti	Musahabeti@gmail.com	739852838
	4	Fizza Zeeshan	Fizzazeeshan123@gmail.com	769232837
	5	Ali Samow	Alisamow@hotmail.com	737963239
	6	John Doe	johndoe@yahoo.com	769748669
	7	Mickey Mouse	mickeymouse88@gmail.com	733336863
*	NULL	NULL	NULL	NULL

INSERT INTO orders (order_id , order_type , order_date, price,
customer_customer_id)

VALUES

('6' , 'Cutlery' , '2020-12-17' , '1500' , '6'),

('7' , 'Health' , '2020-06-02' , '445' , '7');



1 • **SELECT** * **FROM** ecommerce.orders;

<

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cells

	order_id	order_type	order_date	price	customer_customer_id
▶	1	Bakery Items	2021-02-22	250	1
	2	Household Items	2021-01-19	350	2
	3	Clothing	2021-01-02	800	3
	4	Outdoor	2021-02-07	1000	4
	5	Cosmetics	2021-01-11	750	5
	6	Cutlery	2020-12-17	1500	6
	7	Health	2020-06-02	445	7
*	NULL	NULL	NULL	NULL	NULL

orders 1 x Ap

Output

Action Output ▼

	#	Time	Action	Message
✓	1	18:08:15	insert into orders (order_id, order_type, order_date, price, customer_cu...	2 row(s) affected Records
✓	2	18:08:27	SELECT * FROM ecommerce.orders LIMIT 0, 1000	7 row(s) returned

UPDATE

UPDATE shipping_details

SET address = '35 James Street'

WHERE shipping_detailsid = '1';

The screenshot displays a database management interface. The top section shows a SQL query editor with the following query:

```
1 • SELECT * FROM ecommerce.shipping_details;
```

Below the query editor is a 'Result Grid' showing the results of the SELECT query. The grid has five columns: shipping_detailsid, description, shipping_status, address, and orders_order_id. The data is as follows:

shipping_detailsid	description	shipping_status	address	orders_order_id
1	chocolate cake	2 days	11 Kings Street	1
2	fragile	3 days	47 Alivallsgatan	2
3	leather	5 days	31 Mölndal	3
4	DIY	7 days	08 Spannmålsgatan	4
5	red color	1 day	22 Maria Park	5
NULL	NULL	NULL	NULL	NULL

The bottom section shows the same SQL query editor with the following query:

```
1 • update shipping_details
2   SET address = '35 James Street'
3   WHERE shipping_detailsid = '1';
4
5
```

Below the query editor is an 'Output' panel showing the execution results. The output is as follows:

#	Time	Action	Message
1	19:45:13	update shipping_details SET address = '35 James Street' WHERE shipping_detailsid = '1'	0 row(s) affected Rows matched: 1 Changed: 0 Warnings: 0

Limit to 1000 rows

```
1 • SELECT * FROM ecommerce.shipping_details;
```

Result Grid

shipping_detailsid	description	shipping_status	address	orders_order_id
1	chocolate cake	2 days	35 James Street	1
2	fragile	3 days	47 Alivallsgatan	2
3	leather	5 days	31 Mölndal	3
4	DIY	7 days	08 Spannmålsgatan	4
5	red color	1 day	22 Maria Park	5
•	NULL	NULL	NULL	NULL

shipping_details 1 x

Output

#	Time	Action	Message
1	19:45:13	update shipping_details SET address = '35 James Street' WHERE shipping_detailsid = ...	0 row(s) affected Rows matched: 1 Changed: 0 Warnings: 0
2	19:45:36	SELECT * FROM ecommerce.shipping_details LIMIT 0, 1000	5 row(s) returned

UPDATE product

SET weight = '11'

WHERE product_id = '4';

1 • SELECT * FROM ecommerce.product;

	name	product_id	quantity	weight	orders_order_id
▶	Chocolate Cake	1	2	1	1
	Tea Cups	2	6	2	2
	Jacket	3	1	4	3
	Garden Chairs	4	4	10	4
	Lipstick	5	3	3	5
*	NULL	NULL	NULL	NULL	NULL

1 • update product
2 set weight = '11'
3 where product_id = '4';
4

Output

#	Time	Action	Message
1	19:48:29	update product set weight = '11' where product_id = '4'	0 row(s) affected Rows matched: 1 Changed: 0 Warnings: 0

	name	product_id	quantity	weight	orders_order_id
▶	Chocolate Cake	1	2	1	1
	Tea Cups	2	6	2	2
	Jacket	3	1	4	3
	Garden Chairs	4	4	11	4
	Lipstick	5	3	3	5
*	NULL	NULL	NULL	NULL	NULL

product 1 ×

Output

📄 Action Output ▼







	#	Time	Action
✓	1	19:48:29	update product set weight = '11' where product_id = '4'
✓	2	19:48:41	SELECT * FROM ecommerce.product LIMIT 0, 1000
✓	3	19:48:59	SELECT * FROM ecommerce.product LIMIT 0, 1000

DELETE

DELETE FROM shipping_details

WHERE description = 'DIY';

```
1 • SELECT * FROM ecommerce.shipping_details;
```

Result Grid					
Filter Rows: <input type="text"/>					
Edit:   					
Export/Import:  					
Wrap Cell Content: 					
shipping_detailsid	description	shipping_status	address	orders_order_id	
1	chocolate cake	2 days	35 James Street	1	
2	fragile	3 days	47 Alivallsgatan	2	
3	leather	5 days	31 Mölndal	3	
4	DIY	7 days	08 Spannmålsgratan	4	
5	red color	1 day	22 Maria Park	5	
NULL	NULL	NULL	NULL	NULL	

Limit to 1000 rows

Jump to

```
1 • SELECT * FROM ecommerce.shipping_details;
```

<
>

Result Grid
Filter Rows:
Edit:
Export/Import:
Wrap Cell Content:

	shipping_detailsid	description	shipping_status	address	orders_order_id
▶	1	chocolate cake	2 days	35 James Street	1
	2	fragile	3 days	47 Alivallsgatan	2
	3	leather	5 days	31 Mölndal	3
	5	red color	1 day	22 Maria Park	5
•	NULL	NULL	NULL	NULL	NULL

shipping_details 1 x
Apply
Revert
Context Help
Snippets

Output

Action Output

#	Time	Action	Message	Duration / F
✓ 1	20:14:27	DELETE FROM shipping_details WHERE description = 'DIY'	0 row(s) affected	0.000 sec
✓ 2	20:14:39	SELECT * FROM ecommerce.shipping_details LIMIT 0, 1000	4 row(s) returned	0.000 sec

Limit to 1000 rows

Jump to

```
1 • DELETE FROM shipping_details
2 WHERE description = 'DIY';
3
```

Output

Action Output

#	Time	Action	Message
✓ 1	20:14:27	DELETE FROM shipping_details WHERE description = 'DIY'	0 row(s) affected

BASIC/INNER JOINS

SELECT customer.customer_name AS 'Customer Names' , orders.order_id AS 'Order ID'

FROM customer inner join orders

ON customer.customer_id = orders.customer_customer_id;

The screenshot shows a database query tool interface. At the top, there's a tab labeled 'E-commerce Queries'. Below it is a toolbar with various icons. The main area displays a SQL query:

```
31
32 • SELECT customer.customer_name AS 'Customer Names' , orders.order_id AS 'Order ID'
33 FROM customer inner join orders
34 ON customer.customer_id = orders.customer_customer_id;
35
```

Below the query, there's a 'Result Grid' section. It shows a table with two columns: 'Customer Names' and 'Order ID'. The table contains 7 rows of data:

Customer Names	Order ID
Michael Roos	1
Julia Poulsen	2
Musa Habeti	3
Fizza Zeeshan	4
Ali Samow	5
John Doe	6
Mickey Mouse	7

Below the result grid, there's a 'Result 2' tab. It shows an 'Output' section with a table of action output:

#	Time	Action	Message
1	18:32:11	SELECT customer.customer_name AS 'Customer Names' , orders.order_id AS 'Order ID'	7 row(s) returned

SELECT orders.order_type AS 'Order Type', product.name AS 'Product Name'

FROM orders, product, orders_has_product

WHERE orders.order_id = orders_has_product.orders_order_id AND
product.product_id = orders_has_product.product_product_id;

The screenshot shows a database query tool interface. At the top, a tab labeled 'E-commerce Queries' is active. Below the tab is a toolbar with various icons and a 'Limit to 1000 rows' dropdown. The SQL query is displayed in a text area, with line numbers 35 through 40 on the left. The query is as follows:

```
35
36 • SELECT orders.order_type AS 'Order Type', product.name AS 'Product Name'
37 FROM orders, product, orders_has_product
38 WHERE orders.order_id = orders_has_product.orders_order_id AND product.product_id = orders_has_product.product_product_id;
39
40
```

Below the query, there is a 'Result Grid' section. It includes a 'Filter Rows:' input field, an 'Export:' button, and a 'Wrap Cell Content:' checkbox. The grid displays two columns: 'Order Type' and 'Product Name'. The data is as follows:

Order Type	Product Name
Bakery Items	Chocolate Cake
Household Items	Tea Cups
Clothing	Jacket
Outdoor	Garden Chairs
Cosmetics	Lipstick

At the bottom, there is a 'Result 4' tab with an 'Output' section. It shows a table with columns: '#', 'Time', 'Action', 'Message', and 'Duration'. The first row of data is as follows:

#	Time	Action	Message	Duration
1	18:32:34	SELECT orders.order_type AS 'Order Type', product.name AS 'Product Name' FR...	5 row(s) returned	0.016 se

SELECT orders.order_type, shipping_details.description,
shipping_details.shipping_status

FROM orders

INNER JOIN shipping_details ON orders.order_id =
shipping_details.orders_order_id;

The screenshot shows a database query tool interface. At the top, a tab labeled "E-commerce Queries" is active. Below the tab is a toolbar with various icons and a "Limit to 1000 rows" dropdown. The main area displays a SQL query:

```
39
40 • SELECT orders.order_type, shipping_details.description, shipping_details.shipping_status
41 FROM orders
42 INNER JOIN shipping_details ON orders.order_id = shipping_details.orders_order_id;
43
```

Below the query, there is a "Result Grid" section. It includes a "Filter Rows:" input field, an "Export:" button, and a "Wrap Cell Content:" dropdown. The grid displays the following data:

	order_type	description	shipping_status
▶	Bakery Items	chocolate cake	2 days
	Household Items	fragile	3 days
	Clothing	leather	5 days
	Cosmetics	red color	1 day

At the bottom, there is an "Output" section with a dropdown menu set to "Action Output". It displays a table with the following data:

#	Time	Action	Message
✓ 1	18:37:20	SELECT orders.order_type, shipping_details.description, shipping_details.shipping_...	4 row(s) returned

5. CONCLUSION

I have created an E-commerce database for this project which contains several different relationships, attributes, entities, and different keys.

Overall, the project was good. I did not find any difficulty in making ER, EER diagrams or in writing queries.

6. REFERENCES

I created the ER diagram on this website.

(2015)

7. APPENDIX